NAME (as it appears on your UF ID): _____
(Please **PRINT**)

UF Student ID#: _____

---------- CEN 6070 Software Testing & Verification ----------

Exam 1 -- Summer 2012

You have 90 minutes to work on this exam. It is a "closed-book/closed-notes" test. Pay attention to point values, since you may not have time to work all 19 problems. PRINT your name above NOW and sign the pledge at the bottom of the last page, if appropriate, when you are finished. PLEASE PRINT ANSWERS IN THE SPACE PROVIDED ONLY – PREFERABLY USING A BALLPOINT PEN TO INCREASE LEGIBILITY. Good luck!

1. (3 pts.) What, according to Myers ("The Psychology and Economics of Program Testing"), is "the primary cause for poor program testing"? (Circle ONE only.)

   a. Failing to fully appreciate the creative and intellectually challenging aspects of testing.

   b. Failing to design tests for invalid and unexpected, as well as valid and expected, input conditions.

   c. The failure to employ BOTH *data driven* (i.e., black-box) and *logic-driven* (i.e., white-box) testing techniques.

   d. Inadequate time and resources allocated to testing: most efforts are halted *before* the program has been shown to be error free.

   e. Using an incorrect definition of the word "testing".

2. (8 pts.) Myers describes two types of "exhaustive testing": *exhaustive input testing* and *exhaustive path testing.* Circle either "true" or "false" for each of the following statements concerning these two approaches. To compensate for random guessing, you will receive +2 pts. for each correct answer and -2 pts. for each incorrect answer, with a minimum possible score of 0 pts.

   a. According to Myers, "exhaustive input testing" is a *black-box* testing strategy while "exhaustive path testing" is a *white-box* testing strategy. true false

   b. BOTH "exhaustive input testing" AND "exhaustive path testing" are necessary criteria to find all errors in a program. true false

   c. "Exhaustive input testing" alone would not ensure that every *data-sensitivity error* would be detected. true false

   d. "Exhaustive path testing" alone would not ensure that errors associated with the *absence* of necessary paths in a program would be detected. true false

3. (4 pts.) In "Making Meetings Work for Everybody," Gause and Weinberg suggest two possible reasons why some projects may need to hold what seems to be too many meetings. Briefly describe them.

4. (4 pts.) Which, if any, of the following process control techniques are explicitly described by Fagan in his paper, "Design and code inspections to reduce errors in program development". (Circle one only).

   a. Using inspection results to identify which modules contain the highest error density.

   b. Using historical error detection efficiency data together with initial inspection results to estimate the number of errors remaining in modules.

   c. Using initial *testing* results to identify the most error prone code and inspecting this code before continuing with testing.

   d. (a and b ONLY)

   e. (a, b, and c)

   f. (None of the techniques above were explicitly described in the Fagan paper.)

5. (6 pts.) Recall the equivalence classes identified for variables *"Res?"* and *"Gross_Pay"* in connection with the City Tax Specification:

   - ***Res?***
     - { yes } (V)
     - { no } (V)
     - { other } (I)
   - ***Gross_Pay***
     - [0, 30K] (V)
     - (30K,50K] (V)
     - (50K, MAX] (V)
     - < 0 (I)
     - > MAX (I)

   How many test cases would be required to achieve "***Strong*** *Equivalence Class Testing"* based on this model? How many would be required to achieve "***Weak*** *Equivalence Class Testing"?*

   # of test cases required for ***strong*** *equivalence class testing:* _____

   # of test cases required for ***weak*** *equivalence class testing:* _____

6. (10 pts.) The following statements relate to Grady and Van Slack, "Key Lessons in Achieving Widespread Inspection Use." Indicate whether each is true or false. To compensate for random guessing, you will receive +2 pts. for each correct answer and -2 pts. for each incorrect answer, with a minimum possible score of 0 pts.

a. Grady and Van Slack introduce their paper with a personal anecdote about how a small home heating system inspection fee had resulted in a 100 to 1 return on investment for one of the authors and his spouse.	true	false
b. Unlike many large companies, HP benefited from the adoption of inspection technology across all its divisions in a single, carefully coordinated step.	true	false
c. The inspection process step that varies the most at HP is the rework/follow-up step.	true	false
d. Much of HP's success was due to the premise that moderator certification should be based on strict standardization of the inspection process across all its divisions.	true	false
e. Among other things, a *Chief Moderator* owns the inspection process, gathers and reports statistics across all inspections, and drives inspection process improvements.	true	false

7. (4 pts.) Consider the following pseudocode program (which appeared in the "Prerequisite Knowledge Self-Assessment Pre-Test" for this course):

```
Y := X
if X<0 then
    Y := -Y
end_if_then
```

Suppose X and Y are integer variables with initial values X_0 and Y_0.

a. Describe what the program *does* (i.e., what its *function* is) in words and give the *final* values of the variables it references in terms of their *initial* values.

b. Is the program *correct?* Briefly explain your answer.

8. In their paper, "The Effectiveness of Software Development Technical Reviews: A Behaviorally Motivated Program of Research," Sauer, et al., report that 79% of a sample of 29 authors and managers believe *synergy* is a reason why *(group)* software development technical reviews are conducted.

   a. (2 pts.) What, according to the authors, do behavioral studies suggest about ***synergy*** as a source of group performance advantage *in general?*

   b. (2 pts.) What does the theory predict for software development technical reviews regarding the performance advantage of group meetings ***in discovering new defects*** beyond the aggregation of those discovered by individuals?

   c. (3 pts.) Does the theory predict a performance advantage of group meetings based on any other consideration associated with software development technical reviews? If so, identify the consideration and explain.

9. (8 pts.) Provide a SIMPLE counter-example which proves that Path Coverage does NOT subsume Compound Condition Coverage. (Do not attempt to prove the converse.) EXPLAIN HOW YOUR COUNTER-EXAMPLE PROVES THIS.

10. As discussed in class, (machine-based) testing usually begins with *functional* (black-box) tests, is (then) supplemented by *structural* (white-box) tests, and progresses from the unit level toward the system level with one or more integration steps.

    a. (3 pts.) Briefly describe the *rationale* discussed in class for running black box tests before designing and implementing white-box tests.

    b. (3 pts.) Aside from the issue of *when* program elements may become available for testing, what is the *principal advantage* of testing "that progresses from the unit level toward the system level" over testing at the system level only?

11. (4 pts.) What, exactly, is "soak testing," and what is its purpose?

12. (3 pts.) Suppose 60 errors are seeded into a program. After 2.5 days of testing, 20 of the seeded errors have been detected, together with 10 non-seeded errors. Using the technique discussed in class, estimate the number of ***remaining*** non-seeded errors in the program.

13. Recall the Cause-Effect (C-E) Analysis test case selection (coverage criterion) strategy know as "Strategy #3":

> **REPEAT**
>
> Select the next (initially, the first) Effect.
>
> Tracing back through the graph (right to left), find **all feasible combinations of connected Cause values that result in the Effect being True.**
>
> For each **new** such combination found:
>
> Determine values of all other Effects, and
>
> Enter values for each Cause and Effect in a new column of the test case coverage matrix.
>
> **UNTIL** each Effect has been selected.

a. (3 pts.) Which one of the following correctly characterizes the general relationship between Strategy #3 and AFCCV ("All Feasible Combinations of Cause Values")? (Circle one only.)

i. Strategy #3 subsumes AFCCV

ii. AFCCV subsumes Strategy #3

iii. Strategy #3 and AFCCV are independent (neither criterion subsumes the other)

iv. Strategy #3 and AFCCV are equivalent (each criterion subsumes the other)

v. (none of the above)

b. (3 pts.) Suppose for a general C-E Model with k Effects, we wish to cover All Feasible combinations of **EFFECT** Values (AFC**E**V). If we assume that for any feasible combination of Cause values, at least one Effect must be true, how many test cases, at most, would be required to achieve AFC**E**V? (Circle one only.)

i. 1	v. k+1	ix. 2^k-1
ii. 2	vi. $(k-1)^2$	x. 2^k
iii. k-1	vii. k^2	xi. 2^k+1
iv. k	viii. $(k+1)^2$	xii. infinite

14. (8 pts.) Recall the Matherr "STANDARD CONFORMANCE" table for the pow function.

	DOMAIN	OVERFLOW	UNDERFLOW
usual cases	-----	+/-HUGE_VAL	+/-TINY_VAL
0**0	1	-----	-----
(x<0)**(not int)	0	-----	-----
0**(y<0)	+HUGE_VAL	-----	-----

Briefly describe the behavior of pow **and** Matherr in terms of any value(s) returned and/or variable(s) set for each of the following cases:

a. pow(+HUGE_VAL, 2)

b. pow(0, 0)

c. pow (-3, 2.4)

d. pow (0, -3)

15. The need for *"scaffolding"* was discussed in the context of integration testing.

a. (6 pts.) Briefly describe the three different types of *scaffolding* that may be required during integration testing and the role played by each type.

b. (2 pts.) Is *scaffolding* ever required outside the context of *integration* testing? Briefly explain your answer.

16. (13 pts.) Match each description below to the **SINGLE MOST APPROPRIATE TERM** among the following. (Note: terms may apply to none, one, or more than one description.)

A. Fault-based test	L. "Soak" test
B. Installability test	M. Device and configuration test
C. Thread test	N. Usability test
D. Alpha test	O. Serviceability test
E. Performance test	P. Beta test
F. Stress test	R. Security test
G. "Lights out" test	S. "Smoke" test
H. Regression test	T. Exhaustive test
I. Reliability test	U. Compatibility/conversion test
J. Post-test analysis	W. Causal analysis
K. Benchmarking	X. Test-driven development

___ Typical coverage includes post-delivery change procedures (adaptive, perfective, and corrective scenarios), supporting documentation, and system diagnostic tools

___ End-user testing performed within the user environment prior to general release

___ Issues of focus include login and password procedures and policies, levels of authorization for data or procedural access, etc.

___ Appropriate interpretations for "failure" and "time" are critical

___ Integrating program elements associated with a key program function

___ Process aimed at identifying the origin of errors and approaches to eliminate future occurrences

___ General practice of recording and comparing indices of performance, quality, cost, etc., to help identify "best practices"

___ Testing following the insertion of "errors" (e.g., syntactic mutations) into a program

___ Also known as "build verification"; initial test after a software build to detect catastrophic failure

___ May be automated by combining a keystroke recorder and playback tool with a data/output comparator

___ Focus is on typical requirements that systems exhibit "graceful" failures and non-abrupt performance degradation

___ Automated, stand-alone testing not requiring human involvement

___ Specialized testing in which HCI experts conduct experiments and utilize protocol analysis

17. a. (6 pts.) Provide a 3-node control-flow graph with appropriate node and edge annotations for the pseudocode program below that is suitable for dataflow coverage analysis. The 3 nodes of your graph should correspond to the line numbers given in the pseudocode.

```
1. input(A,B)
   while (A>10) do
2.    A := A-B
   end_while
3. output(A,B)
```

b. (10 pts.) In the table below, list all definition-use pairs for variables A and B, and identify all (and only) the associated du-paths for each pair.

variable	**du-pair**	**du-path(s)**

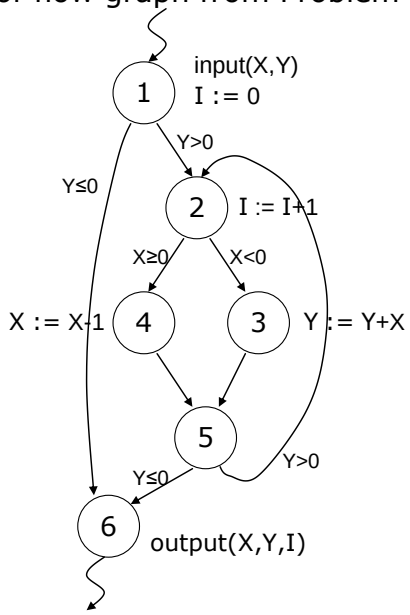
c. (6 pts.) What is the total number of test cases, at the minimum, that would be required to achieve each of the following for this program? (Circle one only for each.)

i. *All-Defs coverage:* 1 2 3 4 8 16 infinite

ii. *All-Uses coverage:* 1 2 3 4 8 16 infinite

iii. *All-DU-Paths coverage:* 1 2 3 4 8 16 infinite

18. (6 pts.) Recall the control-flow graph from Problem Set 3 below.



Give the path condition in terms of X,Y inputs for program path <1,2,3,5,2,3,5,6>. Show ALL path condition predicates; do NOT simplify or combine terms.

19. (3 pts.) Some have suggested that Test-Driven Development (TDD) reflects an element of "methodological hyperbole." ("Hyperbole" in this context, means a deliberate exaggeration used for dramatic effect.). In particular, the process model includes a step that seems to serve no purpose other than to *dramatically reinforce* the principle that testable requirements for new functionality should be the focus BEFORE code is written to implement that functionality -- a principle that is already clearly reflected in the process requirement that a test be written and implemented *before* the functionality is implemented. What is this additional, seemingly "hyperbolic step" in the TDD process model?

On my honor, I have neither given nor received unauthorized aid on this exam and I pledge not to divulge information regarding its contents to those who have not yet taken it.

_____

SIGNATURE